

Web Developers' HTTP Toolbox

Lorna Mitchell, Nexmo



Why HTTP? Why Toolbox?

We build the web, it's made of HTTP.

Let's look at the tools we need.



Getting to Know HTTP

Let's start with some theory

Getting to Know HTTP

HTTP HyperText Transport Protocol

Request

Host	example.com
Path	/index.php
Method	POST
Headers	Accept: text/html, User-Agent: curl, Content-Type: app
Body	fruit=apple&noun=sparkle

Response

Status	200
Headers	Content-Type: text/html
Body	<h1>Hello!</h1>

HTTP Verbs

A verb is a "doing word" ... in HTTP as well as in English.

- GET
- POST
- DELETE
- PUT
- PATCH
- HEAD

HTTP Headers

Request headers:

- Host, Accept

Response headers:

- ETag, Location

Entity headers:

- Content-Type

HTTP Status Codes

Code	Meaning
200	OK
201	Created
204	No Content
302	Found
307	Moved Temporarily

HTTP Status Codes

Code	Meaning
400	Bad Request
401	Unauthorized
403	Forbidden
404	Not Found
500	Server Error

HTTP and PHP

PHP can be client, server, or both

Speaking HTTP

Speaking HTTP

To speak HTTP, you will need a client. Suggestions include:

- curl <https://curl.haxx.se/>
- Postman <https://www.getpostman.com/>
- HTTPie <https://httpie.org/>
- http-console <https://github.com/cloudhead/http-console>
- Paw (Mac only) <https://paw.cloud/client>
- Insomnia <https://insomnia.rest/>
- probably your IDE?

Exercise 1: Get the Sample App

Data Formats

Form encoded:

```
Content-Length: 40
```

```
Content-Type: application/x-www-form-urlencoded
```

```
message=This is a message&name=lornajane
```

JSON data:

```
Content-Type: application/json
```

```
Content-Length: 52
```

```
{"message": "This is a message", "name": "lornajane"}
```


Working with JSON

Use the tools!

In PHP: `json_encode()` and `json_decode()`. No string matching and definitely no regex.

At the commandline:

- `jq` <https://stedolan.github.io/jq/>
- `fx` <https://github.com/antonmedv/fx>

Exercise 2: Set up and use your HTTP client

Making Requests

Making Requests

PHP can also be an HTTP client.

To make HTTP requests:

- use PHP streams
- use Guzzle <http://docs.guzzlephp.org/en/stable/>
- your favourite framework may also have specific features

HTTP With Streams

Make an HTTP request with PHP, no dependencies.

```
$url = "https://httpbin.org/post";  
$data = ["name" => "lornajane", "message" => "Hi there"];  
  
$context = stream_context_create(  
    ["http" => [  
        "method" => "POST",  
        "content" => http_build_query($data)  
    ]  
]);  
  
$response = file_get_contents($url, false, $context);
```

HTTP With Guzzle

Use guzzlehttp\guzzle from Composer

```
require "vendor/autoload.php";

$url = "https://httpbin.org/post";
$data = ["name" => "lornajane", "message" => "Hi there"];

$client = new \GuzzleHttp\Client();
$response = $client->request("POST", $url, [
    "form_params" => $data
]);
```

HTTPBin

<https://httpbin.org> is a site where you can make requests and get information about the responses.

- `/post` used above shows me the incoming post data and how it was parsed, other verbs also available
- `/response/*` offers a selection of canned responses, such as json data, robots.txt file
- `/status/*` returns the specified status code - great for testing error cases

RequestBin

Open source project you can run locally (Docker) or deploy (Heroku) to receive and diagnose any HTTP request.

<https://github.com/runscope/requestbin> (but try my lornajane fork for some fixes)

Useful for testing outgoing API requests, webhooks, anything really!